Dice.py

When rolling a six-sided die, there are six possible outcomes: 1, 2, 3, 4, 5, and 6. If the die is fair, each of the six outcomes is equally likely. If you roll two six-sided dice and add the results together, the possible outcomes are 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, and 12. Anyone who plays board games such as Monopoly knows that each of these outcomes are not equally likely: 7 is the most common result, and 2s and 12s are very rare.

In this assignment, you will simulate dice rolls using a *pseudorandom number generator*. Python provides a set of built-in functions that generate numbers that are not truly random, but are close enough to random for our purposes.

To simulate the roll of two dice, you need to do the following:

- Insert the statement `import random` at the top of your program. This gives your program access to the Python built-in functions for generating pseudorandom numbers.
- The function `random.randint(i,j)` returns a random integer between i and j inclusive, with all possible outcomes occurring *with equal likelihood*. (The italics here are important!) Use this function to generate a random dice roll.

You can force the random number generator to produce the same sequence of random numbers by giving it a *seed* value. You do this by adding the statement `random.seed()` before your first call to `randint`. In order to make it easier for the TA to grade the programs, please start your main program this way:

```
def main():
   random.seed(1314)
```

This will result in everyone getting the same "random" dice rolls. (This is another reason why these numbers are called "pseudorandom" numbers: they can't really be random if they can be reproduced.)

Your program should ask the user to enter the number of trials to perform; that is, the number of times it should try rolling the two six-sided dice. After it rolls the dice the required number of times, it should draw a histogram displaying the results.

If you ran your program using a value of 10 for the number of trials, your output should look something like the following:

```
How many times do you want to roll the dice? 10
Results:  [0, 0, 1, 0, 2, 2, 2, 0, 3, 0, 0]

|                           *
|              *  *  *      *
|        *     *  *  *      *
+--+--+--+--+--+--+--+--+--+--+-
   2  3  4  5  6  7  8  9 10 11 12
```

Notes on the output:

- The exact wording of the text on the first line is not important.
- The second line contains a list summarizing the number of each outcome: in the above example, the program rolled no 2s, no 3s, one 4, no 5s, two 6s, two 7s, two 8s, no 9s, three 10s, and no 11s or 12s.
- The histogram should be formatted to look **exactly** like the above:
    - Each row should begin with a vertical bar, representing a piece of the y-axis.
    - The x-axis should be drawn exactly as shown, with plus signs ("+") at the origin and at the bottom of each bar of the graph, with two minus ("-") signs in between, and a trailing minus sign at the end.
    - The x-axis should be labeled with the numbers 2 through 12 spaced **exactly** as shown.
    - Each of the asterisks ("*") in the above example represents one dice roll of the value indicated by the label below it. Your data will not necessarily look like the example above, but rather, will show the appropriate number of asterisks according to the list mentioned above. **This is the only place your histogram should differ from the example.**

The above histogram looks great if the number of trials you're running is somewhere between 1 and 100. But if you run, say, 10000 trials, the height of the graph will exceed the height of the screen. For this reason, the output of your program should be automatically *scaled* based on the number of trials you are running.

In the example below, we ran 1000 trials, 54 of which resulted in a roll of 3. Instead of displaying 54 asterisks in the "3" column, we scaled the output by a factor of 10. 54/10 = 5.4, which we rounded to 5, the number of asterisks that we ultimately displayed.

```
How many times do you want to roll the dice? 1000
Results:  [28, 54, 89, 104, 157, 162, 128, 109, 75, 64, 30]

|                *  *
|                *  *
|                *  *
|                *  *  *
|                *  *  *
|                *  *  *  *
|             *  *  *  *  *
|          *  *  *  *  *  *
|          *  *  *  *  *  *  *
|          *  *  *  *  *  *  *
|          *  *  *  *  *  *  *  *
|      *  *  *  *  *  *  *  *  *
|      *  *  *  *  *  *  *  *  *
|  *  *  *  *  *  *  *  *  *  *  *
|  *  *  *  *  *  *  *  *  *  *  *
|  *  *  *  *  *  *  *  *  *  *  *
+--+--+--+--+--+--+--+--+--+--+--+-
   2  3  4  5  6  7  8  9 10 11 12
```

In general:

- If the number of trials is less than or equal to 100, the number of asterisks should match the number of outcomes exactly.
- If the number of trials is greater than 100, the number of asterisks should be the number of outcomes * (100/trials), rounded to the nearest integer.